Producer/Consumer Problem

Requirements

- Individual lab
- Due: April 28 (Friday), 5:00pm
- We DO NOT use xv6 for this lab
- Place your completed code in directory ~/Lab5 at odin.cslabs.clarkson.edu
- You don’t need to submit the source code to git repository

Problem

- Producer/Consumer Problem
  - Multiple producers fill data in a buffer
  - Multiple consumers remove data from the same buffer
  - Producers need to wait if buffer is full
  - Consumers need to wait if buffer if empty
  - At any time, only one producer or consumer can operate the buffer
- Program two solutions to solve the problem
  - Sample programs for one producer and multiple consumers are given
  - Use Locks and CVs
  - Use Semphores (next week's topic)

Skeleton Code

- pc_cv.c is the skeleton code for the program using locks and CVs
- pc_sem.c is the skeleton code for the one using semphores
- Fill the missing blocks of code as indicated in the files
- When producer x fills an item y from the buffer, print message "**Producer x fills y**"
- When consumer x removes an item from the buffer, print message "**Consumer x removes y**"
- You should be able to gracefully quit the program when clicking "Ctrl+c"
  - The signal handling part has been done in the skeleton code
- You can use rand() to generate a random integer
  - E.g., rand()%100 will generate a random number from 0-100
- If the output is too fast, use sleep() to control the pace

Output Example

```
liu@odin ~/Lab5 $ ./mpmc 20 5 9
Producer 2 fills 63
Consumer 2 removes 63
Producer 4 fills 26
Consumer 3 removes 26
Producer 2 fills 11
Consumer 4 removes 11
Producer 4 fills 29
Consumer 3 removes 29
Producer 2 fills 62
Producer 1 fills 67
Consumer 1 removes 62
Consumer 5 removes 67
^C Stopping...
Producer 0 fills 22
Producer 0 fills -1
Producer 0 fills -1
Producer 0 fills -1
Producer 0 fills -1
```

Demo Time